Tcl8.6.14/Tk8.6.14 Documentation > Tcl Commands, version 8.6.14 > open

Tcl/Tk Applications | Tcl Commands | Tk Commands | [incr Tcl] Package Commands | SQLite3 Package Commands | TDBC Package Commands | tdbc::mysql Package Commands | tdbc::odbc Package Commands | tdbc::postgres Package Commands | tdbc::sqlite3 Package Commands | Thread Package Commands | Tcl C API | Tk C API | [incr Tcl] Package C API | TDBC Package C API

NAME

open — Open a file-based or command pipeline channel

SYNOPSIS

open *fileName*
open *fileName access*
open *fileName access permissions*

DESCRIPTION

This command opens a file, serial port, or command pipeline and returns a channel identifier that may be used in future invocations of commands like **read**, **puts**, and **close**. If the first character of *fileName* is not | then the command opens a file: *fileName* gives the name of the file to open, and it must conform to the conventions described in the **filename** manual entry.

The *access* argument, if present, indicates the way in which the file (or command pipeline) is to be accessed. In the first form *access* may have any of the following values:

r
Open the file for reading only; the file must already exist. This is the default value if *access* is not specified.

r+
Open the file for both reading and writing; the file must already exist.

w
Open the file for writing only. Truncate it if it exists. If it does not exist, create a new file.

w+
Open the file for reading and writing. Truncate it if it exists. If it does not exist, create a new file.

a
Open the file for writing only. If the file does not exist, create a new empty file. Set the file pointer to the end of the file prior to each write.

a+
Open the file for reading and writing. If the file does not exist, create a new empty file. Set the initial access position to the end of the file.

All of the legal *access* values above may have the character **b** added as the second or third character in the value to indicate that the opened channel should be configured as if with the **fconfigure** **-translation binary** option, making the channel suitable for reading or writing of binary data.

In the second form, *access* consists of a list of any of the following flags, most of which have the standard POSIX meanings. One of the flags must be either **RDONLY**, **WRONLY** or **RDWR**.

RDONLY
Open the file for reading only.

WRONLY
Open the file for writing only.

RDWR
: Open the file for both reading and writing.

APPEND
: Set the file pointer to the end of the file prior to each write.

BINARY
: Configure the opened channel with the **-translation binary** option.

CREAT
: Create the file if it does not already exist (without this flag it is an error for the file not to exist).

EXCL
: If **CREAT** is also specified, an error is returned if the file already exists.

NOCTTY
: If the file is a terminal device, this flag prevents the file from becoming the controlling terminal of the process.

NONBLOCK
: Prevents the process from blocking while opening the file, and possibly in subsequent I/O operations. The exact behavior of this flag is system- and device-dependent; its use is discouraged (it is better to use the **fconfigure** command to put a file in nonblocking mode). For details refer to your system documentation on the **open** system call's **O_NONBLOCK** flag.

TRUNC
: If the file exists it is truncated to zero length.

If a new file is created as part of opening it, *permissions* (an integer) is used to set the permissions for the new file in conjunction with the process's file mode creation mask. *Permissions* defaults to 0666.

COMMAND PIPELINES

If the first character of *fileName* is "|" then the remaining characters of *fileName* are treated as a list of arguments that describe a command pipeline to invoke, in the same style as the arguments for **exec**. In this case, the channel identifier returned by **open** may be used to write to the command's input pipe or read from its output pipe, depending on the value of *access*. If write-only access is used (e.g. *access* is "**w**"), then standard output for the pipeline is directed to the current standard output unless overridden by the command. If read-only access is used (e.g. *access* is "**r**"), standard input for the pipeline is taken from the current standard input unless overridden by the command. The id of the spawned process is accessible through the **pid** command, using the channel id returned by **open** as argument.

If the command (or one of the commands) executed in the command pipeline returns an error (according to the definition in **exec**), a Tcl error is generated when **close** is called on the channel unless the pipeline is in non-blocking mode then no exit status is returned (a silent **close** with -blocking 0).

It is often useful to use the **fileevent** command with pipelines so other processing may happen at the same time as running the command in the background.

SERIAL COMMUNICATIONS

If *fileName* refers to a serial port, then the specified serial port is opened and initialized in a platform-dependent manner. Acceptable values for the *fileName* to use to open a serial port are described in the PORTABILITY ISSUES section.

The **fconfigure** command can be used to query and set additional configuration options specific to serial ports (where supported):

-mode *baud*,*parity*,*data*,*stop*
: This option is a set of 4 comma-separated values: the baud rate, parity, number of data bits, and number of stop bits for this serial port. The *baud* rate is a simple integer that specifies the connection speed. *Parity* is one of the following letters: **n**, **o**, **e**, **m**, **s**; respectively signifying the parity options of "none", "odd", "even", "mark", or "space". *Data* is the number of data bits and should be an integer from 5 to 8, while *stop* is the number of stop bits and should be the integer 1 or 2.

-handshake *type*
: (Windows and Unix). This option is used to setup automatic handshake control. Note that not all handshake types maybe supported by your operating system. The *type* parameter is case-independent.

: If *type* is **none** then any handshake is switched off. **rtscts** activates hardware handshake. Hardware handshake signals are described below. For software handshake **xonxoff** the handshake characters can be redefined with **-xchar**. An additional hardware handshake **dtrdsr** is available only under Windows. There is no default handshake configuration, the initial value depends on your operating system settings. The **-handshake** option cannot be queried.

-queue
(Windows and Unix). The **-queue** option can only be queried. It returns a list of two integers representing the current number of bytes in the input and output queue respectively.

-timeout *msec*
(Windows and Unix). This option is used to set the timeout for blocking read operations. It specifies the maximum interval between the reception of two bytes in milliseconds. For Unix systems the granularity is 100 milliseconds. The **-timeout** option does not affect write operations or nonblocking reads. This option cannot be queried.

-ttycontrol *{signal boolean signal boolean ...}*
(Windows and Unix). This option is used to setup the handshake output lines (see below) permanently or to send a BREAK over the serial line. The *signal* names are case-independent. **{RTS 1 DTR 0}** sets the RTS output to high and the DTR output to low. The BREAK condition (see below) is enabled and disabled with **{BREAK 1}** and **{BREAK 0}** respectively. It is not a good idea to change the **RTS** (or **DTR**) signal with active hardware handshake **rtscts** (or **dtrdsr**). The result is unpredictable. The **-ttycontrol** option cannot be queried.

-ttystatus
(Windows and Unix). The **-ttystatus** option can only be queried. It returns the current modem status and handshake input signals (see below). The result is a list of signal,value pairs with a fixed order, e.g. **{CTS 1 DSR 0 RING 1 DCD 0}**. The *signal* names are returned upper case.

-xchar *{xonChar xoffChar}*
(Windows and Unix). This option is used to query or change the software handshake characters. Normally the operating system default should be DC1 (0x11) and DC3 (0x13) representing the ASCII standard XON and XOFF characters.

-pollinterval *msec*
(Windows only). This option is used to set the maximum time between polling for fileevents. This affects the time interval between checking for events throughout the Tcl interpreter (the smallest value always wins). Use this option only if you want to poll the serial port more or less often than 10 msec (the default).

-sysbuffer *inSize*

-sysbuffer *{inSize outSize}*
(Windows only). This option is used to change the size of Windows system buffers for a serial channel. Especially at higher communication rates the default input buffer size of 4096 bytes can overrun for latent systems. The first form specifies the input buffer size, in the second form both input and output buffers are defined.

-lasterror
(Windows only). This option is query only. In case of a serial communication error, **read** or **puts** returns a general Tcl file I/O error. **fconfigure** **-lasterror** can be called to get a list of error details. See below for an explanation of the various error codes.

SERIAL PORT SIGNALS

RS-232 is the most commonly used standard electrical interface for serial communications. A negative voltage (-3V..-12V) define a mark (on=1) bit and a positive voltage (+3..+12V) define a space (off=0) bit (RS-232C). The following signals are specified for incoming and outgoing data, status lines and handshaking. Here we are using the terms *workstation* for your computer and *modem* for the external device, because some signal names (DCD, RI) come from modems. Of course your external device may use these signal lines for other purposes.

TXD(output)
Transmitted Data: Outgoing serial data.

RXD(input)
Received Data: Incoming serial data.

RTS(output)
Request To Send: This hardware handshake line informs the modem that your workstation is ready to receive data. Your workstation may automatically reset this signal to indicate that the input buffer is full.

CTS(input)
Clear To Send: The complement to RTS. Indicates that the modem is ready to receive data.

DTR(output)
: **Data Terminal Ready:** This signal tells the modem that the workstation is ready to establish a link. DTR is often enabled automatically whenever a serial port is opened.

DSR(input)
: **Data Set Ready:** The complement to DTR. Tells the workstation that the modem is ready to establish a link.

DCD(input)
: **Data Carrier Detect:** This line becomes active when a modem detects a "Carrier" signal.

RI(input)
: **Ring Indicator:** Goes active when the modem detects an incoming call.

BREAK
: A BREAK condition is not a hardware signal line, but a logical zero on the TXD or RXD lines for a long period of time, usually 250 to 500 milliseconds. Normally a receive or transmit data signal stays at the mark (on=1) voltage until the next character is transferred. A BREAK is sometimes used to reset the communications line or change the operating mode of communications hardware.

ERROR CODES (Windows only)

A lot of different errors may occur during serial read operations or during event polling in background. The external device may have been switched off, the data lines may be noisy, system buffers may overrun or your mode settings may be wrong. That is why a reliable software should always **catch** serial read operations. In cases of an error Tcl returns a general file I/O error. Then **fconfigure -lasterror** may help to locate the problem. The following error codes may be returned.

RXOVER
: Windows input buffer overrun. The data comes faster than your scripts reads it or your system is overloaded. Use **fconfigure -sysbuffer** to avoid a temporary bottleneck and/or make your script faster.

TXFULL
: Windows output buffer overrun. Complement to RXOVER. This error should practically not happen, because Tcl cares about the output buffer status.

OVERRUN
: UART buffer overrun (hardware) with data lost. The data comes faster than the system driver receives it. Check your advanced serial port settings to enable the FIFO (16550) buffer and/or setup a lower(1) interrupt threshold value.

RXPARITY
: A parity error has been detected by your UART. Wrong parity settings with **fconfigure -mode** or a noisy data line (RXD) may cause this error.

FRAME
: A stop-bit error has been detected by your UART. Wrong mode settings with **fconfigure -mode** or a noisy data line (RXD) may cause this error.

BREAK
: A BREAK condition has been detected by your UART (see above).

PORTABILITY ISSUES

Windows
: Valid values for *fileName* to open a serial port are of the form **com***X*, where *X* is a number, generally from 1 to 9. A legacy form accepted as well is **com***X***:**. This notation only works for serial ports from 1 to 9. An attempt to open a serial port that does not exist or has a number greater than 9 will fail. An alternate form of opening serial ports is to use the filename **//./comX**, where X is any number that corresponds to a serial port.

: When running Tcl interactively, there may be some strange interactions between the real console, if one is present, and a command pipeline that uses standard input or output. If a command pipeline is opened for reading, some of the lines entered at the console will be sent to the command pipeline and some will be sent to the Tcl evaluator. If a command pipeline is opened for writing, keystrokes entered into the console are not visible until the pipe is closed. These problems only occur because both Tcl and the child application are competing for the console at the same time. If the command pipeline is started from a script, so that Tcl is not accessing the console, or if the command pipeline does not use standard input or output, but is redirected from or to a file, then the above problems do not occur.

: Files opened in the "**a**" mode or with the **APPEND** flag set are implemented by seeking immediately before each write, which is not an atomic operation and does not carry the guarantee of strict appending that is present on POSIX platforms.

Unix

Valid values for *fileName* to open a serial port are generally of the form **/dev/tty***X*, where *X* is **a** or **b**, but the name of any pseudo-file that maps to a serial port may be used. Advanced configuration options are only supported for serial ports when Tcl is built to use the POSIX serial interface.

When running Tcl interactively, there may be some strange interactions between the console, if one is present, and a command pipeline that uses standard input. If a command pipeline is opened for reading, some of the lines entered at the console will be sent to the command pipeline and some will be sent to the Tcl evaluator. This problem only occurs because both Tcl and the child application are competing for the console at the same time. If the command pipeline is started from a script, so that Tcl is not accessing the console, or if the command pipeline does not use standard input, but is redirected from a file, then the above problem does not occur.

See the **PORTABILITY ISSUES** section of the **exec** command for additional information not specific to command pipelines about executing applications on the various platforms

EXAMPLES

Open a file for writing, forcing it to be created and raising an error if it already exists.

```
set myNewFile [open filename.txt {WRONLY CREAT EXCL}]
```

Open a file for writing as a log file.

```
set myLogFile [open filename.log "a"]
fconfigure $myLogFile -buffering line
```

Open a command pipeline and catch any errors:

```
set fl [open "| ls this_file_does_not_exist"]
set data [read $fl]
if {[catch {close $fl} err]} {
    puts "ls command failed: $err"
}
```

Open a command pipeline and read binary data from it. Note the unusual form with “|[list” that handles non-trivial edge cases with arguments that potentially have spaces in.

```
set fl [open |[list create_image_data $input] "rb"]
set binData [read $fl]
close $fl
```

SEE ALSO

file, **close**, **filename**, **fconfigure**, **gets**, **read**, **puts**, **exec**, **pid**, **fopen**

KEYWORDS

access mode, append, create, file, non-blocking, open, permissions, pipeline, process, serial